International Journal of Trendy Research in Engineering and Technology
Volume 4 Issue 7 December 2020
ISSN NO 2582-0958

DETECTION OF MALICIOUS ANDROID APPS USING MACHINE LEARNING TECHNIQUES

Cline Walter Colaco, Mandar Deepak Bagwe Siddharth Amit Bose
Final Year UG Students, Dept. of computer Engineering , Xavier Institute of Engineering, Mumbai - 400016,

Received 21 November 2020 Received in revised form 03 December 2020 Accepted 04 December 2020
Available online 15 December 2020

ABSTRACT

Lately, the matter of dangerous malware in devices is spreading speedily, particularly those repackaged android malware. Though understanding robot malware mistreatment dynamic analysis will give a comprehensive read, it's still subjected to high price in setting preparation and manual efforts within the investigation. Android is the most preferred openly available smart phone OS and its permission declaration access management mechanisms can't sight the behavior of malware. the matter of police investigation such malware presents distinctive challenges thanks to the restricted resources accessible and restricted privileges granted to the user however conjointly presents distinctive opportunities within the needed data hooked up to every application. Through this article, we tend to gift a machine learning-based system for the detection of malware on android devices. In our project, a code behavior signature-based malware detection framework mistreatment associate degree SVM rule is planned, which might sight malicious code and their variants effectively in runtime and extend malware characteristics information dynamically. Experimental results show that the approach incorporates a high detection rate and low rate of false positive and false negative, the power, and performance impact on the first system can even be unheeded. Our system extracts variety of options associate degreed trains a Support Vector Machine in an offline (off-device) manner, so as to leverage the upper computing power of a server or cluster of servers.

Keywords: Malware analysis, Machine Learning, Neural Networks, Support Vector Machine.

INTRODUCTION

Malware is basically a malicious program software or computer code, may be a general term won't to talk over with a range of styles of hostile or intrusive computer code like viruses, worms, spyware, Trojan horses, rootkits, and backdoors. A typical feature of Malware is that it's specifically designed to wreck, disrupt, steal, or normally, impose unhealthy or illegitimate actions. Malware will virtually infect any information processing system running user programs (or applications), and also the propagation and bar of the malware are well studied for private computers. Especially for smart phone devices, current solutions for locating malware within the mobile platform are way behind the pace of the increasing quality of mobile applications. A recent report has shown that there are more than 87 million mobile applications presently obtainable on the market. This quality of the mechanical man system has LED to an enormous increase within the spreading of mechanical man malware. This malware is principally distributed in markets operated by third parties, however even the Google mechanical man Market cannot guarantee that each one of its listed applications area unit threat free. The threats for mechanical man embody Phishing, Banking-Trojans, Spyware, Bots, Root Exploits, SMS Fraud, Premium Dialers, and pretend Installers. There have conjointly been reports regarding Download-Trojans Apps that transfer their malicious code when installation which suggests that these Apps can't be simply detected by Google's technology throughout publication within the Google mechanical man Market.

This paper discusses, portrays and focuses on an SVM-based active learning framework for smart phone malware detection, and within the mechanical man system valid the effectiveness of the strategy, tests show that the planned methodology has sensible relevancy and measurability will be complete on a range of well-liked malware observation and might detect unknown malware. Due to its less impact on system performance, potentially significant impact on the initial system capability may go unnoticed. In summary, malware applications normally use the subsequent 3 sorts of penetration techniques for installation, activation, and running on the android device: Repackaging among the rest of many is the foremost common techniques for malware developers to put in malicious applications on a mechanical man platform. These sorts of approaches commonly begin from well-liked legitimate Apps and misuse them as malware. The developers commonly transfer well-liked Apps, take apart them, add their own malicious codes, so re-assemble and transfer the new App to official or different markets. However, changing the technique through which an application is made and maintained makes it tougher for malware detection. Developers should still use repackaging however rather than

enveloping the impose code within App, they embody Associate in Nursing update element that may transfer malicious code at runtime. Downloading is the ancient attack technique, malware developers desire engaging users, to transfer fascinating and enticing Apps.

PROBLEM DEFINITION

To create an efficient system that curbs the threat of android malware by correctly detecting and mitigating any malicious APKs via combining permissions and API calls as features to characterize malware, and use machine learning techniques to automatically extract patterns to differentiate benign and malicious Apps.

SCOPE

Our software will effectively identify, detect, categorize apps and safeguard android mobile devices from malicious apps thus avoiding any stealing or misuse of the user's data by using an easy user interface.

REVIEW OF LITERATURE

The initial studies on smart phone malware were chiefly targeted on understanding the threats and behaviors of rising malware. There has been vital work on the matter of police work malware on mobile devices. Many approaches monitor the facility usage of applications and report abnormal consumption. Others monitor system calls and arrange to discover uncommon system call patterns. Different approaches use additional ancient comparison with acknowledged malware or different heuristics. Signatures primarily based ways, introduced within the mid-90s area unit ordinarily employed in malware detection. The main weakness of this kind of approach is its weakness in police work metamorphic and unseen malware. Rather than victimization predefined signatures for malware detection, data processing and machine learning techniques give a good thanks to dynamically extract malware patterns. For smart phone-based mobile computing platforms, recent years have witnessed an increasing range of additional sophisticated malware attacks like repackaging. Recent analysis consistently characterizes existing mechanical man malware from varied aspects, together with their installation ways, activation mechanism moreover because the nature of carried malicious payloads. supported the analysis with four representative mobile security software package over 1200 collected malware, their experiments show the weakness of current malware detection solutions and need the necessity to develop next-generation anti-mobile-malware solutions. One existing work has used data processing and options generated from windows workable API calls. They achieved sensible leads to a really giant scale dataset with concerning 35,000 transportable workable files. Another activity foot printing methodology additionally provides a dynamic approach to discover self-propagating malware. All these existing ways have basically advanced the mechanical man malware detection; however the misuse detection isn't reconciling to the novel mechanical man malware and continually needs frequent change of the signatures. Here lies the analysis gap.

In comparison, our work is motivated by a number of the higher than techniques and approaches, however with a spotlight on developing straightforward and effective malware detection approaches, while not looking forward to advanced dynamic runtime analysis and any static predefined malware signatures. Our objective is to mix permissions and API calls as options to characterize malware and use machine learning techniques to mechanically extract patterns to differentiate benign and malicious Apps. This study may be a static analysis that uses the options that may be extracted from the supply codes of the app's .apk files.

DESCRIPTION

ANALYSIS

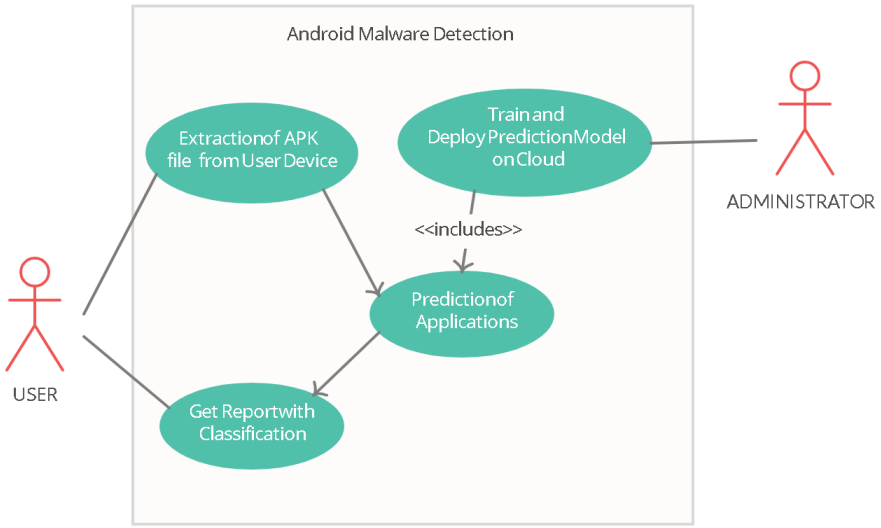

Fig 4.1: Use Case Diagram

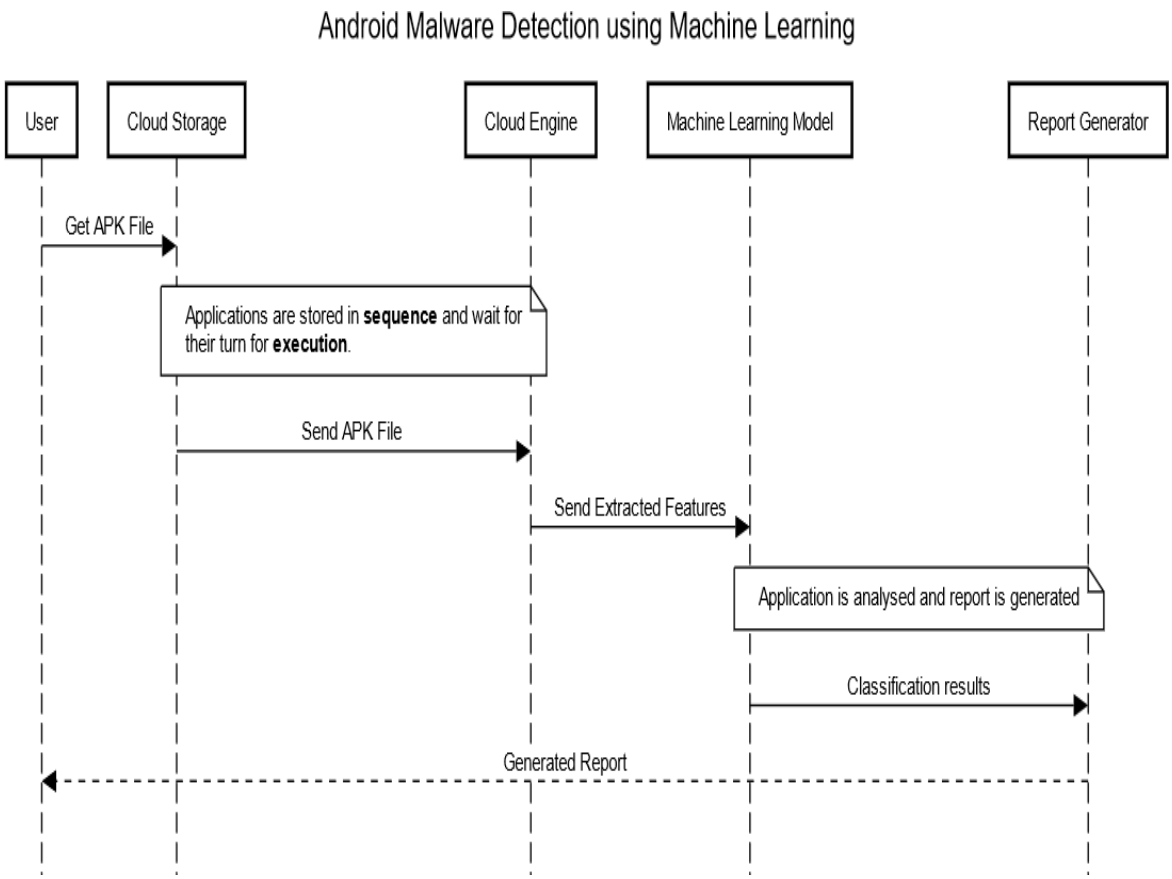

Fig 4.2: Sequence Diagram

The User selects the APK file which is to be tested and it is sent to the cloud storage. Over there the applications are stored in sequence and wait to be executed. The APK is then sent to the Cloud Engine. The features of the APK such as permissions and API calls are extracted and sent to the Machine Learning model. Thus, the features of the APK are analyzed and based on the findings a report is generated and sent back to the User.

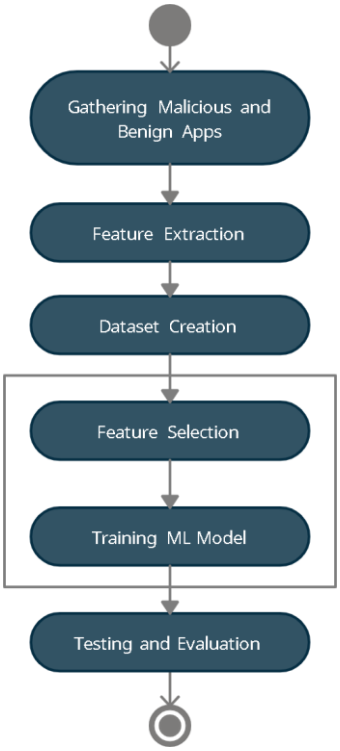

Fig 4.3: Flow of the Project

DESIGN

Firstly, we collect a number of malicious and benign android applications by retrieving their APKs online. Next, we implement feature extraction by enumerating the API calls, permissions and activity of the APKs to get a fair idea of the behavior of the particular apps. Then we create a dataset based on the extracted features by compiling them into CSV files for future use and reference. After that we select specific features which seem like an anomaly or a red flag, collect and store them in new CSV files and train our Machine Learning model according to them. Finally, our last phase is to test our model with the data and evaluate our findings according to the results.

IMPLEMENTATION METHODOLOGY

(Data Collection and Feature Filtering)

1. Collect all applications in separate folders which contain benign as well as suspicious applications respectively.
2. Using "Glob" framework in python create an array of files is for further processing.
3. Analyze each application in the array using "pyaxmlparser" and "androguard" framework.
4. Extract the following things in the analysis phase:

a. Permissions
b. Activities
c. Intents
d. API calls

5. Taking these four attributes into consideration a program maps all attributes to a CSV file and mentions a class for each application.
6. Once CSV files are generated, analyze them for any redundancy present, and if found, eliminate the entire row.
7. Another program extracts the total permissions from these APK files. These permissions will work as attributes in the Dataset CSV File (Here if permission is present it is marked as 1 else it is marked as 0).
8. An N-bit Vector extracts search line in the CSV file, these vectors work as input to the machine learning algorithm.

(Machine Learning)

(SVM)

1. Import data in the form of CSV file using pandas framework.
2. Using train test split divide entire dataset in a ratio of

1:3 (75% of data is for training and 25% for testing).

3. Design an SVM model while keeping the inputs in mind.
4. Select 'rbf' kernel as input/output is binary.
5. Analyze accuracy using confusion matrix.

(KNN)

1. Import data in form of CSV file using pandas framework.
2. Using train test split divide entire dataset in a ratio of 1:3 (75% of data is for training and 25% for testing).
3. Design a KNN model keeping input in mind.
4. Use this model along with a Decision tree.
5. Use The Decision tree for further bifurcation of malware families.

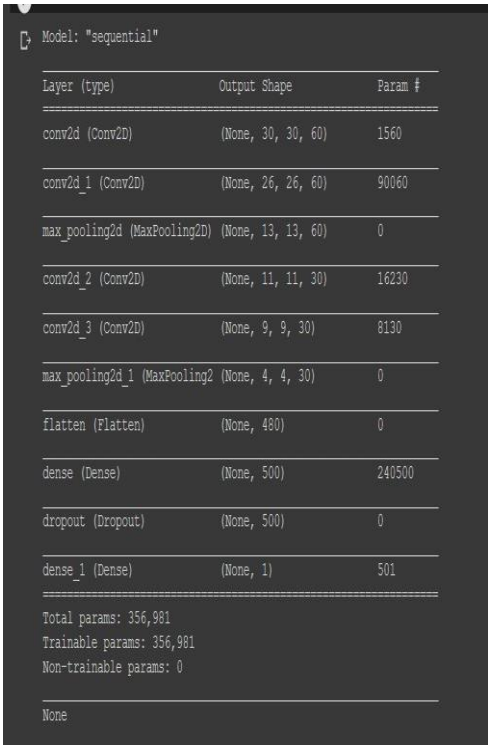

Fig 4.4: Model

These are the statistics of our model that we have implemented.

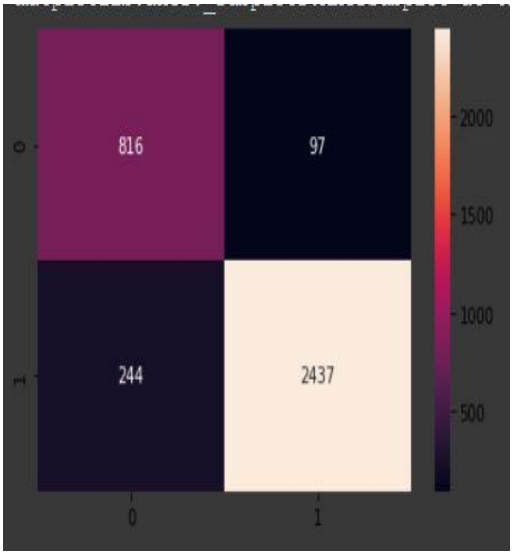

Fig 4.5: Confusion Matrix

DETAILS OF HARDWARE AND SOFTWARE

1) Hardware Details

- Cloud Engine
- Cloud Storage
- Android Device

2) Software Details

- Python 3.x
- Tensorflow 2.2.x
- Android Studio 5

Future Scope

Once model training is complete the next step is to deploy the entire solution on a cloud/server. Then the next step that we have in plan is to make an android app which would make a backup of the application which is to be tested and upload it to cloud. At the cloud end this app is stored as blob storage and these apps would be then processed one by one according to their position in the database. After completion of the analysis a report would be generated stating whether the app is safe or malicious and this would be sent to the user's mobile device. Furthermore if the app is found to be malicious, the report will pinpoint as to why the app is malicious by displaying the specific anomalies found in the permissions, API calls, intent, etc.

CONCLUSION

Hence, we have successfully proposed to use permissions and API calls of Android applications to detect malware and malicious codes in Android based mobile platform. Ours is a novel approach to distinguish and detect Android malware with different intentions. It is effective, that is, it is able to distinguish variant of Android malware between distinct purposes of them. The proposed framework extracts permissions from Android applications and further combines the API calls to characterize each application as a high dimension feature vector. By applying learning methods to the collected datasets, we can derive classification models to classify Apps as benign or malware. Experiments on real world data demonstrate the good performance of the framework for malware detection.

APPENDIX

SVM-Support vector machine
UI-User Interface
API-Application Programming Interface
KNN- K nearest neighbors
CSV-Comma Separated Values
RBF-Radial basis function
APK-Android Application Package

REFERENCES

[1] Androzoo Dataset, https://androzoo.uni.lu/

[2] CIC Dataset 2020, https://www.unb.ca/cic/datasets/maldroid-2020.html

[3] VirusShare.com Dataset, https://virusshare.com/

[4] SandDroid-An automatic Android application analysis system. http://sanddroid.xjtu.edu.cn:8080/#home

[5] Shabtai, A., Kanonov, U., Elovici, Y. et al. "Andromaly": a behavioral malware detection framework for android devices. J Intell Inf Syst 38, 161–190 (2012).

[6] Justin Sahs and Latifur Khan.2012.A Machine Learning Approach to Android Malware Detection. In Proceedings of the 2012 European Intelligence and Security Informatics Conference (EISIC'12). IEEE Computer Society, USA, 141–147.

[7] Zhao M., Ge F., Zhang T., Yuan Z. (2011) AntiMalDroid: An Efficient SVM-Based Malware Detection Framework for Android. In: Liu C, Chang J, Yang A (eds) In- formation Computing and Applications. ICICA2011. Communications in Computer and Information Science, vol 243. Springer, Berlin, Heidelberg.

[8] Seung-Hyun Seo, Aditi Gupta, Asmaa Mohamed Sallam, Elisa Bertino, Kangbin Yim, Detecting mobile malware threats to homeland security through static analysis, Journal of Network and Computer Applications, Volume 38, 2014, Pages 43-53, ISSN 1084-8045,

[9] Arp, Daniel & Spreitzenbarth, Michael & Hubner, Malte & Gascon, Hugo & Rieck, Konrad.(2014). DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket. Symposium on Network and Distributed System Security (NDSS).

[10] D. Wu, C. Mao, T. Wei, H. Lee and K. Wu," DroidMat: Android Malware Detection through Manifest and API Calls Tracing," 2012 Seventh Asia Joint Conference on Information Security, Tokyo, 2012, pp. 62-69, DOI: 10.1109/AsiaJCIS.2012.18. https://ieeexplore.ieee.org/document/6298136/

[11] William Enck, Machigar Ongtang, and Patrick McDaniel.2009. On light weight mobile phone application certification. In Proceedings of the 16th ACM conference on Computer and communications security CCS'09). Association for Computing Machinery, New York, NY, USA, 235–245.

[12] Sanz, Borja & Santos, Igor & Laorden, Carlos & Ugarte Pedrero, Xabier & Bringas, Pablo.

(2012). On the Automatic Categorization of Android Applications. 10.1109/CCNC.2012.6181075.

[13] H.Wang, J.Si, H.Li and Y.Guo," RmvDroid: Towards A Reliable Android Malware Dataset with App Metadata," 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR), Montreal, QC, Canada, 2019, pp.404-408, DOI: 10.1109/MSR.2019.00067.

[14] X. Li, J. Liu, Y. Huo, R. Zhang and Y. Yao, "An Android malware detection method based on Android Manifest file," 2016 4th International Conference on Cloud Computing and Intelligence Systems(CCIS), Beijing, 2016, pp.239-243, DOI: 10.1109/CCIS.2016.7790261.

[15] Mohammed K. Alzaylaee, Suleiman Y. Yerima, Sakir Sezer, DL-Droid: Deep Learning Based Android Malware Detection Using Real Devices, Computers & Security (2019), DOI:

[16] N. Peiravian and X. Zhu," Machine Learning for Android Malware Detection Using Permission and API Calls,"2013 IEEE 25th International Conference on Tools with Artificial Intelligence, Herndon, VA, 2013, pp. 300-305, DOI:10.1109/ICTAI.2013.53.

[17] The Complete Android Oreo Developer Course - Build 23 Apps! Created by Rob Percival, Nick Walter, https://www.udemy.com/course/the-complete-android-oreo-developer-course/